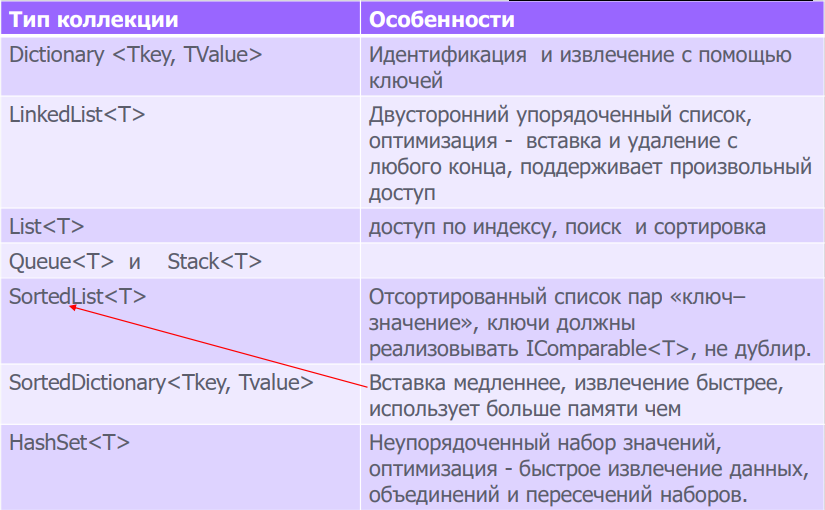
1. На какие основные виды/типы делятся все коллекции .NET? Охарактеризуйте  
   каждый из них.
2. Что такое generic-коллекции? Назовите примеры известных вам generic коллекций.

Обобщенные коллекции — это те же самые обобщенные классы. И опять же их использование перед необобщенными коллекциями имеет те же преимущества: ***повышение производительности*** (не надо тратить время на упаковку и распаковку объекта) и ***повышенная типобезопасность***.

Классы обобщенных коллекций находятся в пространстве имен **System.Collections.Generic**. Функционал коллекций также по большей части описывается в обобщенных интерфейсах.

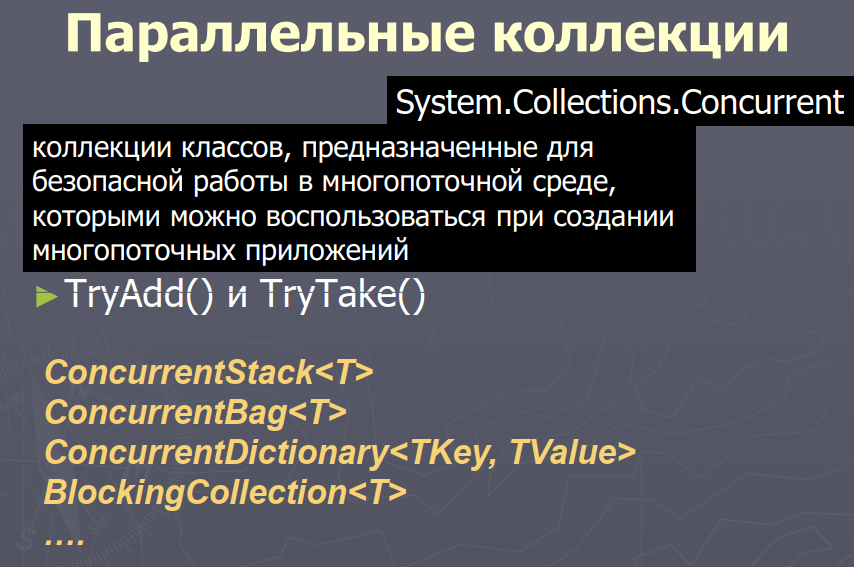
* **List<T>**: класс, представляющий последовательный список.
* **Dictionary<TKey, TValue>**: класс коллекции, хранящей наборы пар "ключ-значение".
* **LinkedList<T>**: класс двухсвязного списка.
* **Queue<T>**: класс очереди объектов, работающей по алгоритму FIFO ("первый вошел - первый вышел").
* **(HashSet<T>) и SortedSet<T>**: Оба они реализуют интерфейс ISet<T>. Класс HashSet<T> содержит неупорядоченный список различающихся элементов, а в SortedSet<T> элементы упорядочены.
* **SortedList<TKey, TValue>**: класс коллекции, хранящей наборы пар "ключ-значение", отсортированных по ключу.
* **SortedDictionary<TKey, TValue>**: класс коллекции, хранящей наборы пар "ключ-значение", отсортированных по ключу. В общем похож на класс SortedList<TKey, TValue>, основные отличия состоят лишь в использовании памяти и в скорости вставки и удаления
* **Stack<T>**: класс стека однотипных объектов.

Большинство обобщенных классов коллекций дублируют необобщенные классы коллекций. Но если вам не надо хранить объекты разных типов, то предпочтительнее использовать обобщенные коллекции.

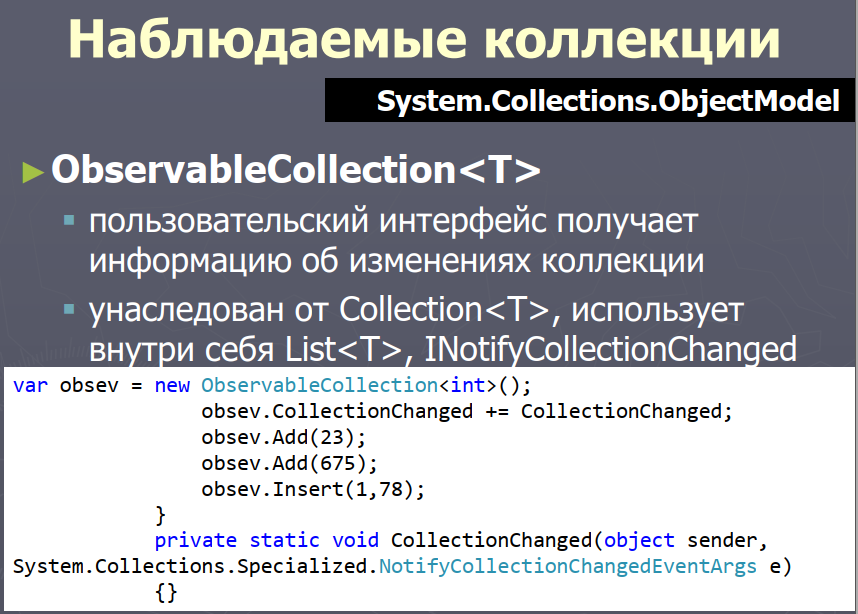
1. В чем разница между ArrayList и Array?

Массив **(System.Array)** фиксируется по размеру после его выделения. Нельзя добавлять к нему элементы или удалять элементы из него. Кроме того, все элементы должны быть одного типа. В результате он безопасен по типу и также является наиболее эффективным, как с точки зрения памяти, так и производительности.

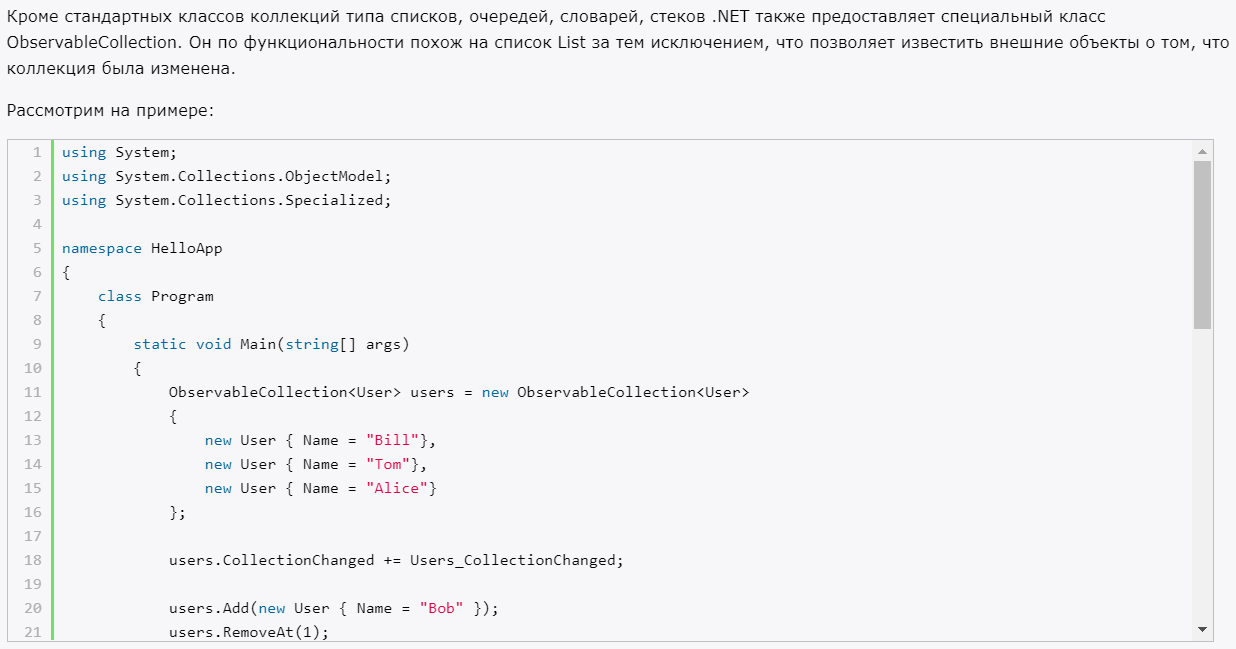
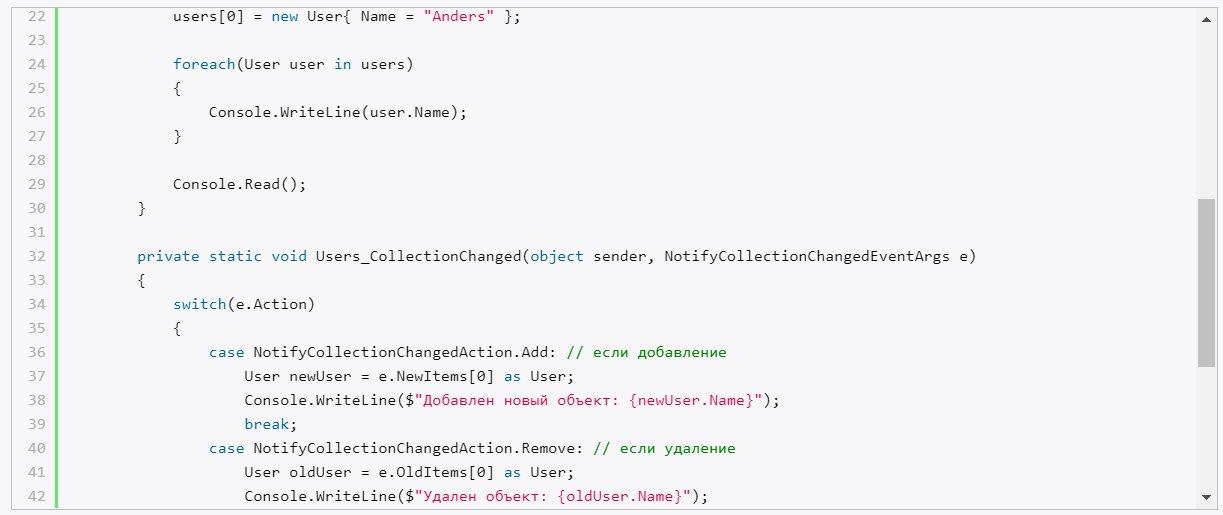
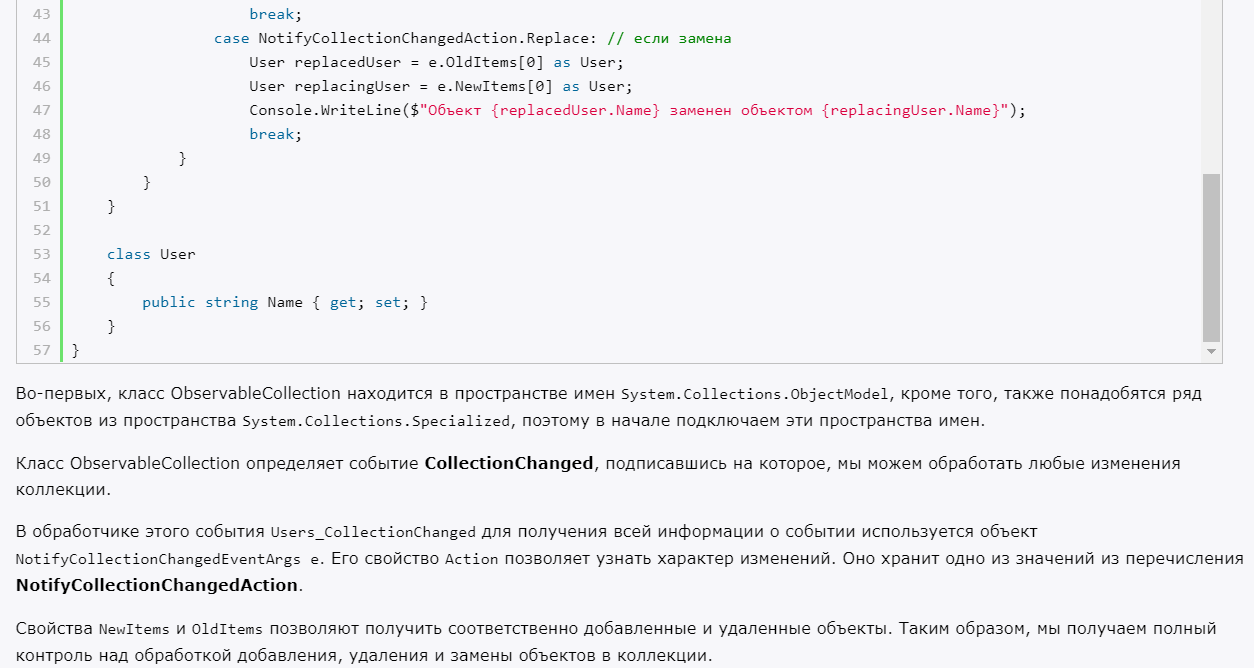
**ArrayList** — это гибкий массив, содержащий список объектов. Можно добавлять и удалять элементы из него, и он автоматически распределит пространство. Если сохранять в нем типы значений, они помещаются в коробку и распаковываются, что может быть немного неэффективным. Кроме того, он не является безопасным для типа.

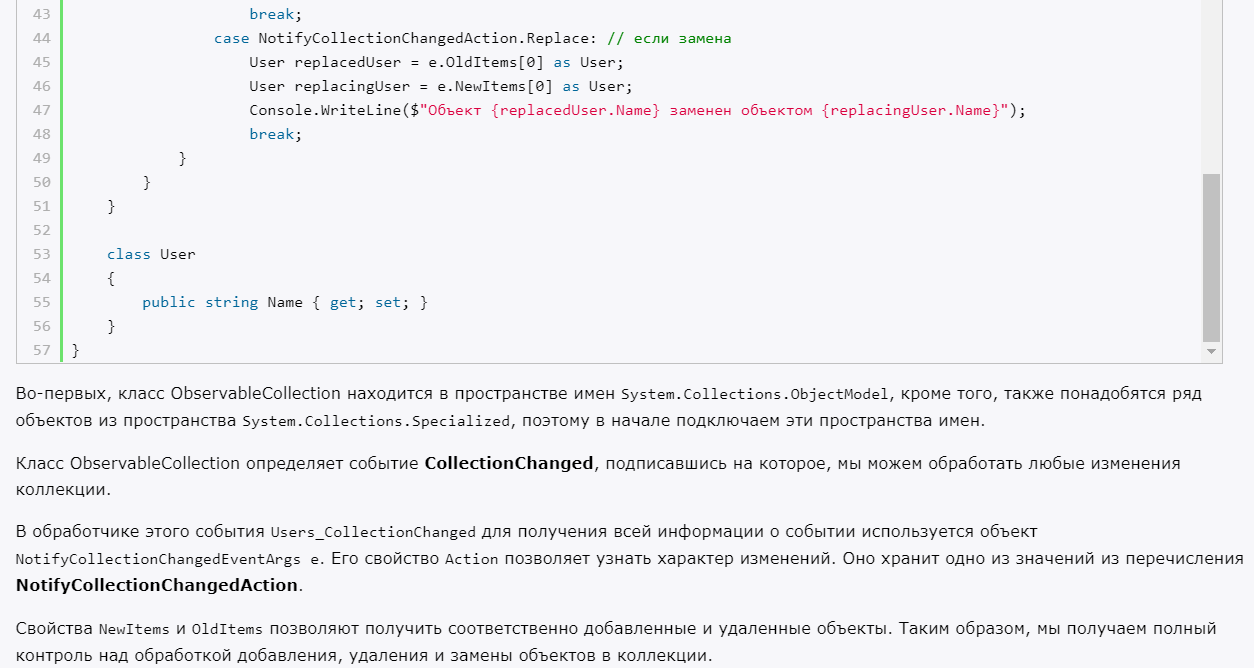
1. Охарактеризуйте коллекции, которые вы использовали в своем варианте.
2. Чем отличаются коллекции, расположенные в пространстве имен  
   System.Collections.Concurrent?
3. Какое пространство имен необходимо подключить в проект, чтобы иметь  
   возможность использовать generic-коллекции?

**System.Collections.Generic**

1. Что такое наблюдаемая коллекция? Как ее можно использовать?

В случае если нужна информация о том, когда элементы коллекции удаляются или добавляются, можно использовать класс ObservableCollection<T>. Этот класс был определен для WPF и предназначен для того, чтобы *пользовательский интерфейс мог получать информацию об изменениях коллекции*.

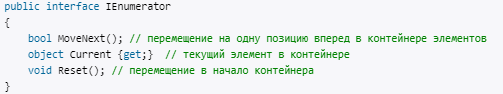




1. Охарактеризуйте интерфейсы IEnumerator, IEnumerable. В чем отличие  
   назначений интерфейсов IEnumerator и IEnumerable.

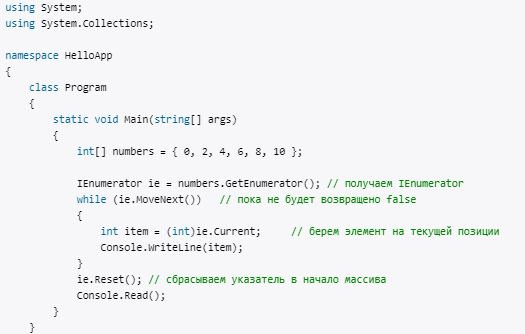
IEnumerator, IEnumerable – **перечислители**.

**IEnumerator -** определяет функционал для перебора внутренних объектов в контейнере

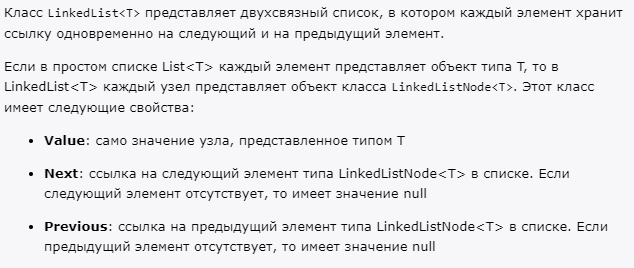
****

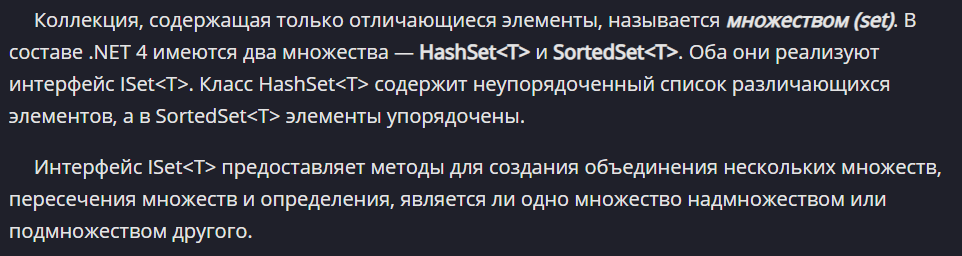
**IEnumerable -** имеет метод, возвращающий ссылку на другой интерфейс - перечислитель

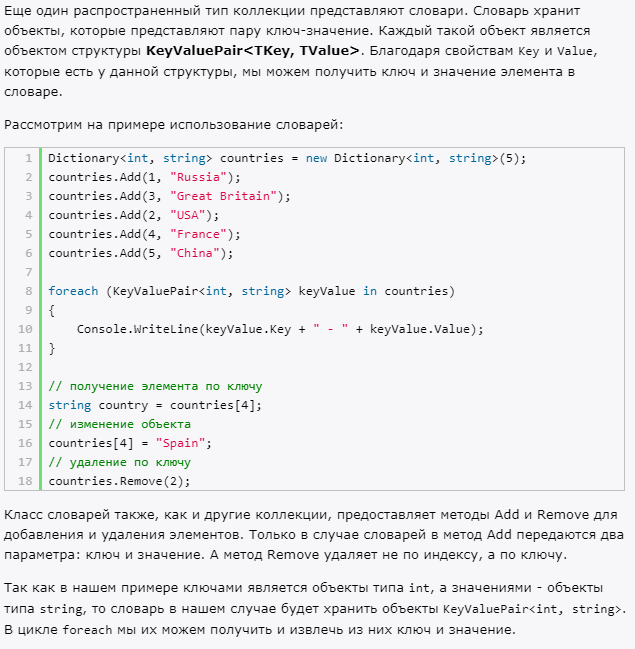




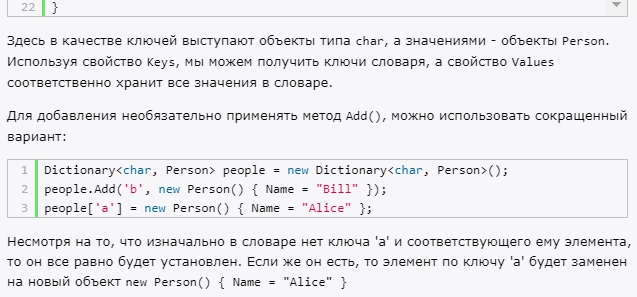
1. Поясните принцип работы коллекций:
   1. LinkedList <T>

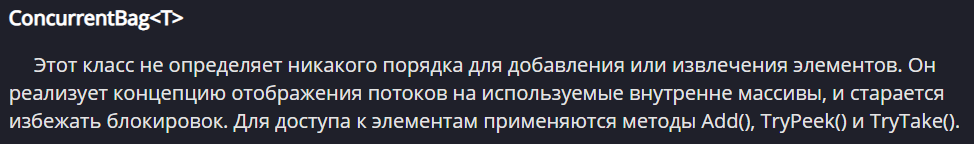


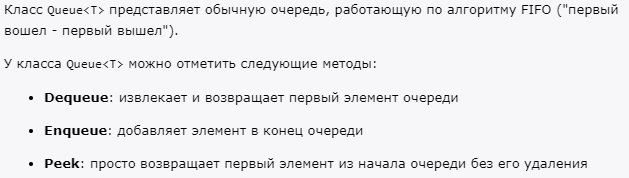
* 1. HashSet <T>
  2. Dictionary <Tkey, TValue>

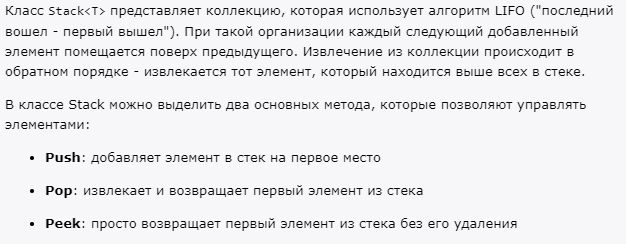




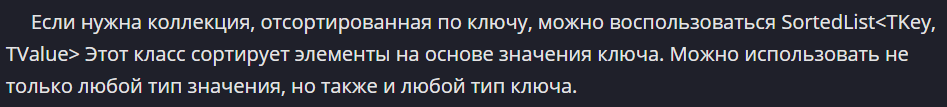


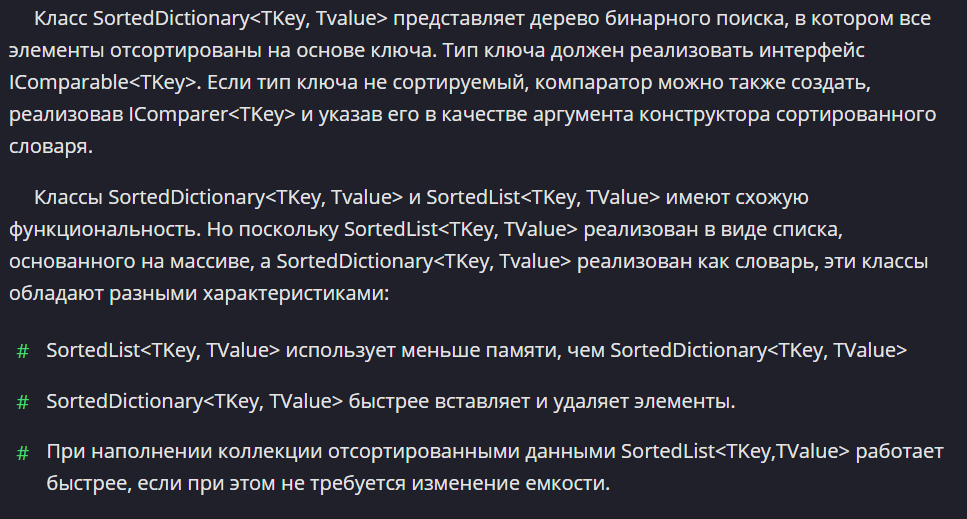
* 1. ConcurrentBag <Tkey, TValue>
  2. Stack<T>, Queue<T>





* 1. SortedList, SortedDictionary

****

****